



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/870,087	05/30/2001	Navin Kumar Sinha	JP920010012US1	6782
39903	7590	04/19/2005	EXAMINER	
ANTHONY ENGLAND PO Box 5307 AUSTIN, TX 78763-5307			YIGDALL, MICHAEL J	
			ART UNIT	PAPER NUMBER
			2192	
DATE MAILED: 04/19/2005				

Please find below and/or attached an Office communication concerning this application or proceeding.

## Office Action Summary

Application No.

09/870,087

Applicant(s)

SINHA, NAVIN KUMAR

Examiner

Michael J. Yigdall

Art Unit

2192

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --  
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

### Status

- 1) ☒ Responsive to communication(s) filed on 16 November 2004.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

### Disposition of Claims

- 4) ☒ Claim(s) 1-20 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) ☐ Claim(s) \_\_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 1-20 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

### Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on \_\_\_\_\_ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

### Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some \* c) ☐ None of:
- ☐ Certified copies of the priority documents have been received.
  - ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
  - ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

### Attachment(s)

- |   |   |
|---|---|
| 1) <input checked="" type="checkbox"/> Notice of References Cited (PTO-892)             | 4) <input type="checkbox"/> Interview Summary (PTO-413)                     |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948)    | Paper No(s)/Mail Date. _____  |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) | 5) <input type="checkbox"/> Notice of Informal Patent Application (PTO-152) |
| Paper No(s)/Mail Date _____   | 6) <input type="checkbox"/> Other: _____                                    |

4

### **DETAILED ACTION**

1. Applicant's amendment and response filed on November 16, 2004 has been fully considered. Claims 1-20 are pending.

#### ***Response to Arguments***

2. Applicant's arguments with respect to claims 1, 2, 11 and 12 have been considered but are moot in view of the new ground(s) of rejection.
3. Applicant contends that the added limitations of claims 1, 2, 11 and 12 patentably distinguish the present invention over the cited teachings of Mahadevan (Applicant's remarks, page 9, second paragraph). However, Mahadevan in view of Ayers, now made of record, teaches the new limitations, as set forth in the claim rejections below.

#### ***Claim Rejections - 35 USC § 103***

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

5. Claims 1-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pat. No. 5,797,013 to Mahadevan et al. (art of record, "Mahadevan") in view of U.S. Pat. No. 5,857,105 to Ayers et al. (art made of record, "Ayers").

With respect to claim 1 (currently amended), Mahadevan discloses a method of optimizing compiled code generated from high level computer programming languages, wherein the compiled code includes loop constructs (see, for example, the abstract), the method comprising the steps:

(1) providing a non-optimized loop code segment corresponding to a loop construct written in a high level programming language, wherein in the non-optimized loop code segment the loop construct is executed a loop repetition number of times  $n$  (see, for example, column 6, lines 22-27, which shows providing loop code for a loop written in a high-level language, and column 7, lines 30-35, which shows that the loop is iteratively executed some number of times, i.e.  $n$  times).

Mahadevan does not expressly disclose the limitation wherein the non-optimized loop code segment includes a call to a procedure, the call depending on a number of arguments, wherein the call invokes the procedure only if a certain condition is met and wherein the certain condition includes one of the arguments being less than another one of the arguments.

However, Ayers discloses an analogous method of optimizing compiled code generated from high-level computer programming languages (see, for example, column 3, lines 12-22), and discloses a non-optimized code segment that includes a call to a procedure that depends on a number of arguments (see, for example, column 4, lines 51-58). Ayers further discloses that the call invokes the procedure only if a certain condition is met in which an argument is less than another argument (see, for example, column 5, lines 10-18).

Mahadevan also discloses the steps:

Art Unit: 2192

(2) providing execution conditions required to cause execution of the loop construct the loop repetition number of times  $n$  (see, for example, column 7, lines 35-39, which shows providing conditions for executing the loop  $n$  times);

(3) optimizing the non-optimized loop code segment for the execution conditions to provide a consolidated code segment corresponding with the execution conditions for execution of the loop said loop repetition number of times  $n$  (see, for example, column 10, lines 53-61, which shows optimizing the loop to provide code corresponding to the trip count, i.e. the number of repetitions  $n$ ).

Although Mahadevan discloses omitting branches from the consolidated loop code segment to save CPU cycles and reduce branch mispredictions (see, for example, column 6, lines 28-32), Mahadevan does not expressly disclose the limitation wherein the consolidated code includes certain code of the non-optimized loop code segment and omits certain other code of the non-optimized loop code segment and wherein the call is omitted from the consolidated loop code segment if the execution conditions indicate the certain condition is not met.

However, Ayers further discloses in-lining, wherein the consolidated code includes code from the called procedure of the non-optimized code segment and omits the call to the procedure, so as to eliminate call overhead (see, for example, column 3, lines 28-32). The in-lining is performed if the execution conditions indicate that a condition is not met (see, for example, column 5, lines 42-46).

Mahadevan also discloses the steps:

(4) determining whether the consolidated code segment should be executed in preference to the non-optimized loop code segments (see, for example, column 10, lines 46-52, which shows determining the most favorable optimization); and

(5) if said determination is favorable, including the consolidated code segment in optimized code for a program written in the high level programming language (see, for example, column 10, lines 53-61, which shows optimizing the code using the most favorable optimization).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the method of Mahadevan with the features taught by Ayers, wherein the non-optimized loop code segment includes a call to a procedure, and the consolidated code includes certain code of the non-optimized loop code segment but omits the call to the procedure. The combination would have been obvious because one of ordinary skill in the art would have been motivated to eliminate call overhead, such as taught by Ayers, so that the method of Mahadevan could save additional CPU cycles.

With respect to claim 2 (currently amended), Mahadevan discloses a method of optimizing the compiled code generated from high level computer programming languages, wherein the compiled code includes loop constructs (see, for example, the abstract), the method comprising the steps:

(1) providing a non-optimized loop code segment corresponding to a loop construct written in a high level programming language, wherein in the non-optimized loop code segment the loop construct is executed a loop repetition number of times  $n$  (see, for example, column 6, lines 22-27, which shows providing loop code for a loop written in a high-level language, and

Art Unit: 2192

column 7, lines 30-35, which shows that the loop is iteratively executed some number of times, i.e.  $n$  times).

Mahadevan does not expressly disclose the limitation wherein the non-optimized loop code segment includes a call to a procedure, the call depending on a number of arguments, wherein the call invokes the procedure only if a certain condition is met and wherein the certain condition includes one of the arguments being less than another one of the arguments.

However, Ayers discloses an analogous method of optimizing compiled code generated from high-level computer programming languages (see, for example, column 3, lines 12-22), and discloses a non-optimized code segment that includes a call to a procedure that depends on a number of arguments (see, for example, column 4, lines 51-58). Ayers further discloses that the call invokes the procedure only if a certain condition is met in which an argument is less than another argument (see, for example, column 5, lines 10-18).

Mahadevan also discloses the steps:

(2) providing a non-optimized pre-loop code segment corresponding to programming instructions preceding the loop construct, and a non-optimized post-loop code segment corresponding to instructions succeeding the loop construct (see, for example, FIG. 3, which shows instructions preceding and succeeding the loop to be provided as pre-loop and post-loop code, respectively);

(3) providing execution conditions required to cause execution of the loop construct the loop repetition number of times  $n$  (see, for example, column 7, lines 35-39, which shows providing conditions for executing the loop  $n$  times);

(4) revising the non-optimized pre-loop, loop and post-loop code segments to include the execution conditions (see, for example, FIG. 3, which shows including the execution conditions with the pre-loop, loop and post-loop code); and

(5) optimizing the non-optimized pre-loop, loop and post-loop code segments for the execution conditions to provide a consolidated code segment corresponding with the execution conditions for execution of the loop said loop repetition number of times  $n$  (see, for example, column 10, line 53 to column 11, line 8, which shows optimizing the loop to provide code corresponding to the trip count, i.e. the number of repetitions  $n$ , and optimizing the placement of compensation code as pre-loop or post-loop code; also see, for example, FIGS. 5 and 6, which show optimized pre-loop, loop and post-loop code).

Although Mahadevan discloses omitting branches from the consolidated loop code segment to save CPU cycles and reduce branch mispredictions (see, for example, column 6, lines 28-32), Mahadevan does not expressly disclose the limitation wherein the consolidated code includes certain code of the non-optimized code loop segment and omits certain other code of the non-optimized loop code segment and wherein the call is omitted from the consolidated loop code segment if the execution conditions indicate the certain condition is not met.

However, Ayers further discloses in-lining, wherein the consolidated code includes code from the called procedure of the non-optimized code segment and omits the call to the procedure, so as to eliminate call overhead (see, for example, column 3, lines 28-32). The in-lining is performed if the execution conditions indicate that a condition is not met (see, for example, column 5, lines 42-46).

Mahadevan also discloses the steps:



(6) determining whether the consolidated code segment should be executed in preference to the non-optimized code segments (see, for example, column 10, lines 46-52, which shows determining the most favorable optimization); and

(7) if said determination is favorable, including the consolidated code segment in optimized code for a program written in the high level programming language (see, for example, column 10, lines 53-61, which shows optimizing the code using the most favorable optimization).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement the method of Mahadevan with the features taught by Ayers, wherein the non-optimized loop code segment includes a call to a procedure, and the consolidated code includes certain code of the non-optimized loop code segment but omits the call to the procedure. The combination would have been obvious because one of ordinary skill in the art would have been motivated to eliminate call overhead, such as taught by Ayers, so that the method of Mahadevan could save additional CPU cycles.

With respect to claims 3 and 7 (currently amended), Mahadevan also discloses the limitation wherein said determination involves a cost-benefit analysis to determine whether the cost of using the consolidated code segment is reduced by a predetermined threshold compared with not using the consolidated code segment (see, for example, column 10, lines 46-52, which shows determining the most favorable optimization while taking into account the cost of not using the most favorable optimization, and column 10, lines 7-45, which shows thresholds used in the determination).

With respect to claims 4 and 8 (original), Mahadevan also discloses the limitation wherein the inclusion of said consolidated code segment in the optimized code is conditional on the occurrence of the execution conditions (see, for example, column 9, lines 46-50, which shows that the inclusion of the optimized loop in the code is conditional, e.g. on the occurrence of the execution conditions).

With respect to claims 5 and 9 (original), Mahadevan also discloses the limitation wherein said loop constructs includes any one or more of the following loop constructs: for loops, while loops, repeat loops (see, for example, column 2, lines 22-24, which shows that the loops may include for loops, while loops and do loops, i.e. repeat loops).

With respect to claim 6 (original), Mahadevan also discloses the limitation wherein said steps (1) to (5) are repeated a predetermined number of times  $k$ , for values of the loop repetition number  $n$  from 0 to  $k-1$  (see, for example, column 7, lines 30-35, which shows that the operations are repeated for each iteration of the loop, and column 6, lines 22-27, which shows processing all the loops).

With respect to claim 10 (original), Mahadevan also discloses the limitation wherein said steps (1) to (7) are repeated a predetermined number of times  $k$ , for values of the loop repetition number  $n$  from 0 to  $k-1$  (see, for example, column 7, lines 30-35, which shows that the operations are repeated for each iteration of the loop, and column 6, lines 22-27, which shows processing all the loops).

With respect to claim 11 (currently amended), Mahadevan discloses a compiler for optimizing the compiled code generated from high level computer programming languages, wherein the compiled code includes loop constructs, the compiler being embodied on a computer-readable medium (see, for example, column 11, lines 9-22, which shows a compiler for optimizing code with loops generated from a high-level language, and FIG. 1, which shows an associated computer-readable medium). The additional features and limitations of this claim are analogous to the limitations recited in claim 1 (see the rejection of claim 1 above).

With respect to claim 12 (currently amended), Mahadevan discloses a compiler for optimizing the compiled code generated from high level computer programming languages wherein the compiled code includes loop constructs, the compiler being embodied on a computer-readable medium (see, for example, column 11, lines 9-22, which shows a compiler for optimizing code with loops generated from a high-level language, and FIG. 1, which shows an associated computer-readable medium). The additional features and limitations of this claim are analogous to the limitations recited in claim 2 (see the rejection of claim 2 above).

With respect to claims 13 and 17 (currently amended), the limitations of these claims are analogous to the limitations recited in claims 3 and 7 (see the rejections of claims 3 and 7 above).

With respect to claims 14 and 18 (original), the limitations of these claims are analogous to the limitations recited in claims 4 and 8 (see the rejections of claims 4 and 8 above).

With respect to claims 15 and 19 (original), the limitations of these claims are analogous to the limitations recited in claims 5 and 9 (see the rejections of claims 5 and 9 above).

With respect to claim 16 (original), the limitations of this claim are analogous to the limitations recited in claim 6 (see the rejection of claim 6 above).

With respect to claim 20 (original), the limitations of this claim are analogous to the limitations recited in claim 10 (see the rejection of claim 10 above).

### *Conclusion*

6. The prior art made of record and not relied upon is considered pertinent to Applicant's disclosure. U.S. Pat. No. 6,292,940 to Sato discloses a program compile system and its compile method for efficiently compiling a source program including an indirect call for a procedure.

7. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire **THREE MONTHS** from the mailing date of this action. In the event a first reply is filed within **TWO MONTHS** of the mailing date of this final action and the advisory action is not mailed until after the end of the **THREE-MONTH** shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than **SIX MONTHS** from the date of this final action.

Art Unit: 2192

8. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Michael J. Yigdall whose telephone number is (571) 272-3707.

The examiner can normally be reached on Monday through Friday from 7:30am to 4:00pm.

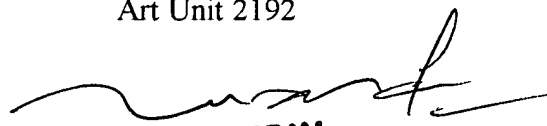
If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

MY

Michael J. Yigdall  
Examiner  
Art Unit 2192

mjy

  
**TUAN DAM**  
**SUPERVISORY PATENT EXAMINER**